FIG. 1

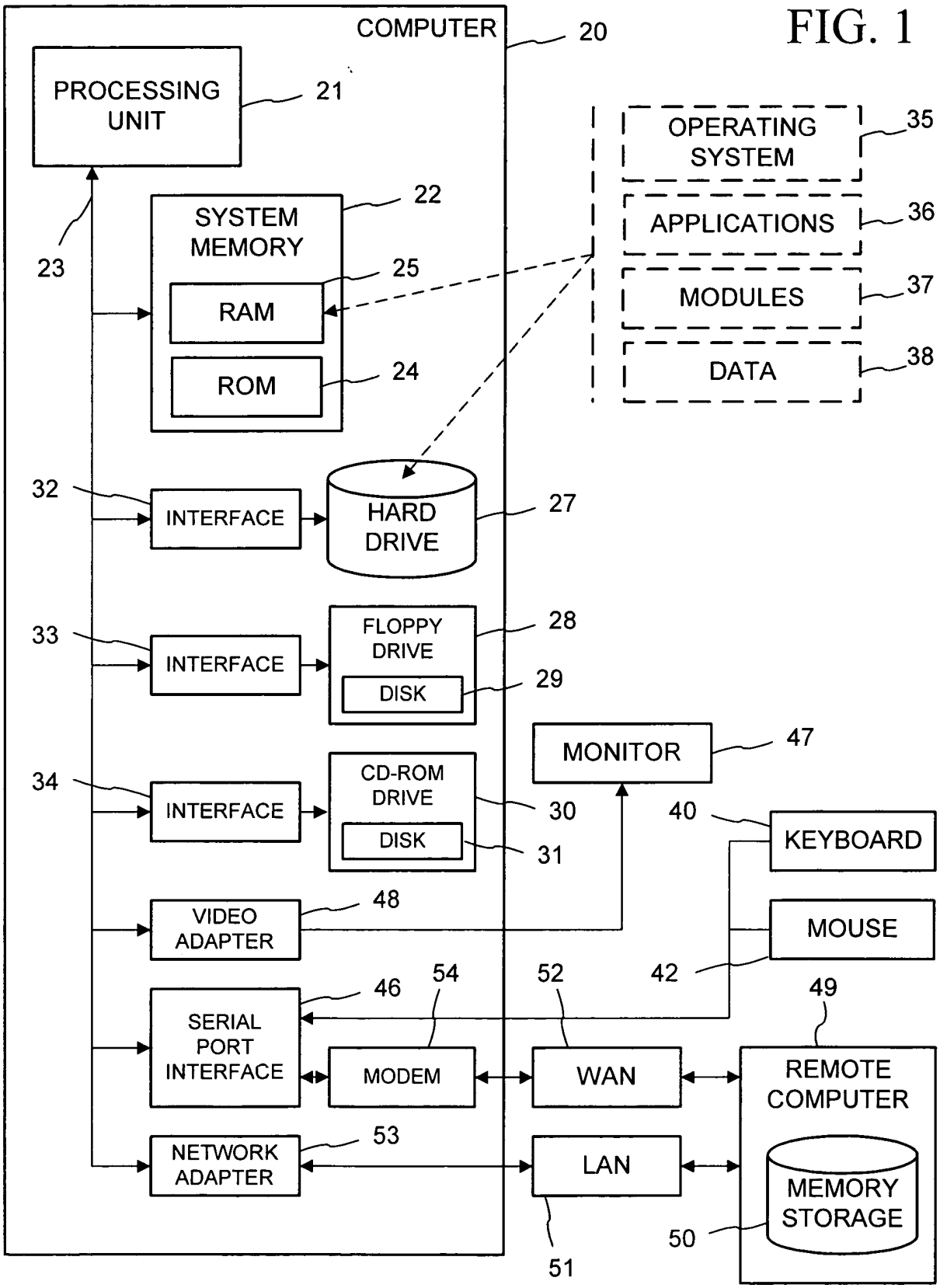

COMPUTER
20
PROCESSING UNIT
21
SYSTEM MEMORY
22
23
RAM
25
ROM
24
OPERATING SYSTEM
35
APPLICATIONS
36
MODULES
37
DATA
38
32
INTERFACE
HARD DRIVE
27
33
INTERFACE
FLOPPY DRIVE
28
DISK
29
34
INTERFACE
CD-ROM DRIVE
30
DISK
31
MONITOR
47
40
KEYBOARD
VIDEO ADAPTER
48
MOUSE
42
46
54
52
49
SERIAL PORT INTERFACE
MODEM
WAN
REMOTE COMPUTER
NETWORK ADAPTER
53
LAN
MEMORY STORAGE
50
51

FIG. 2

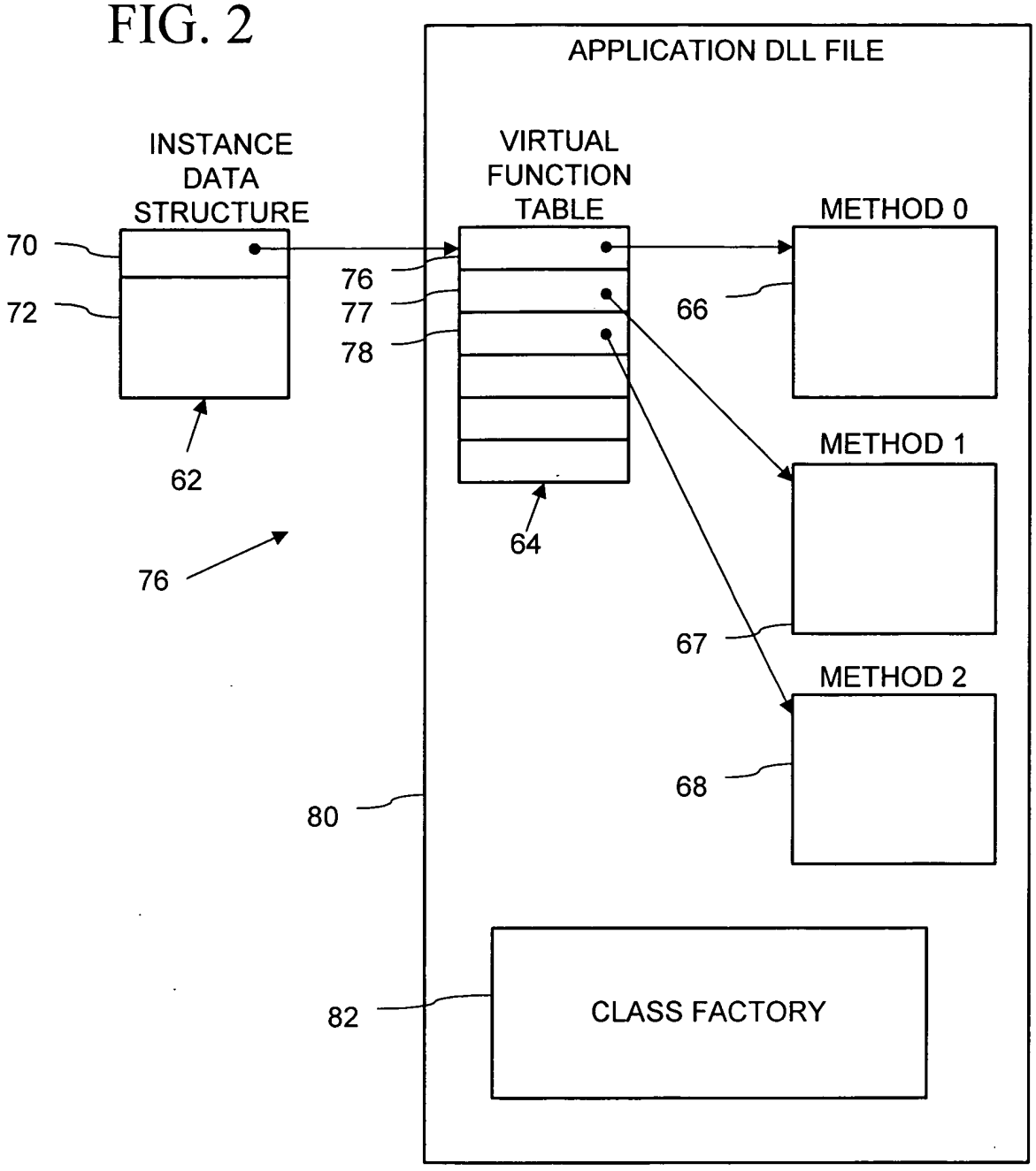

APPLICATION DLL FILE
INSTANCE DATA STRUCTURE
VIRTUAL FUNCTION TABLE
METHOD 0
METHOD 1
METHOD 2
CLASS FACTORY
70
72
62
76
76
77
78
64
66
67
68
80
82

FIG. 3

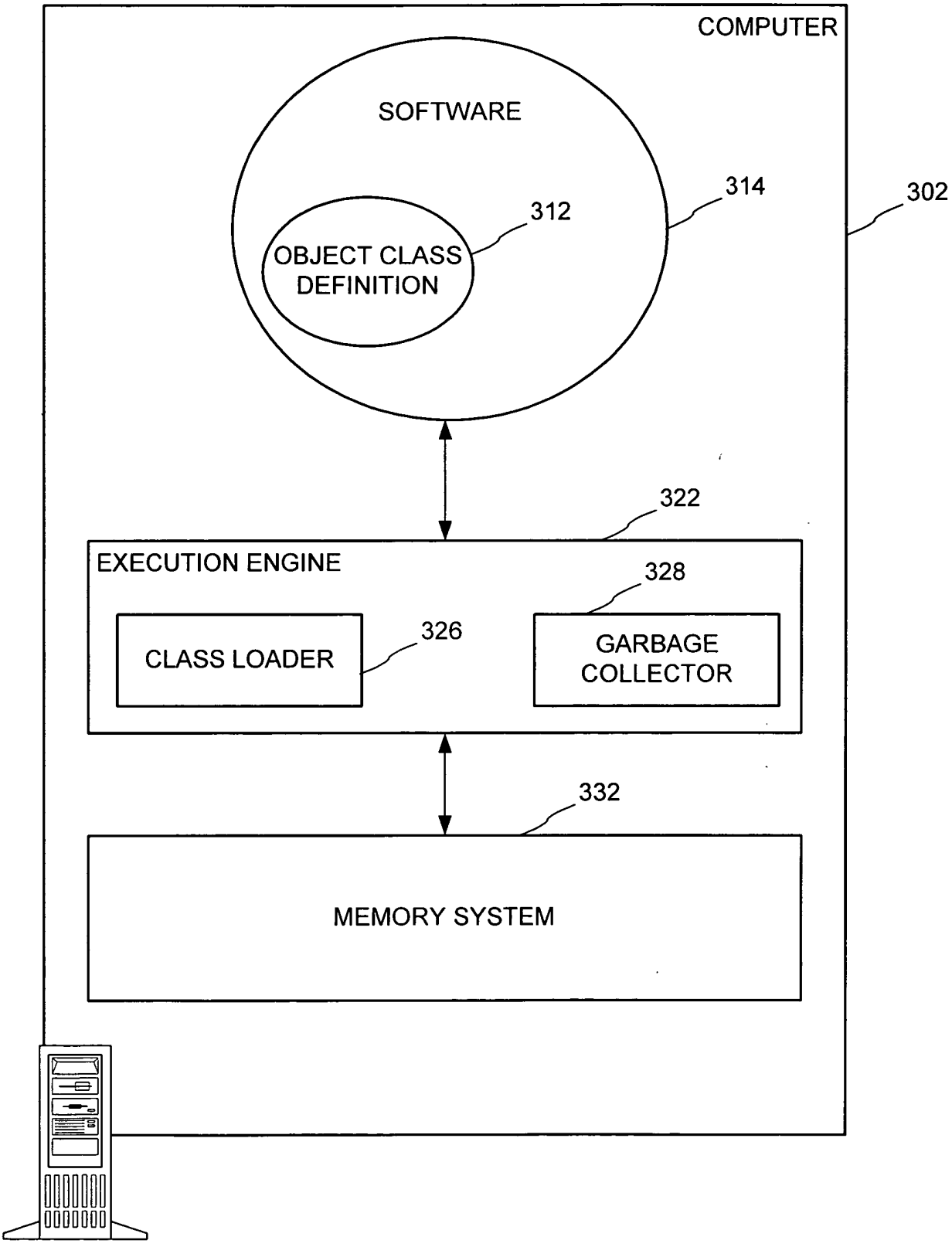

COMPUTER
SOFTWARE
314
302
312
OBJECT CLASS DEFINITION
322
EXECUTION ENGINE
328
CLASS LOADER
326
GARBAGE COLLECTOR
332
MEMORY SYSTEM

FIG. 4

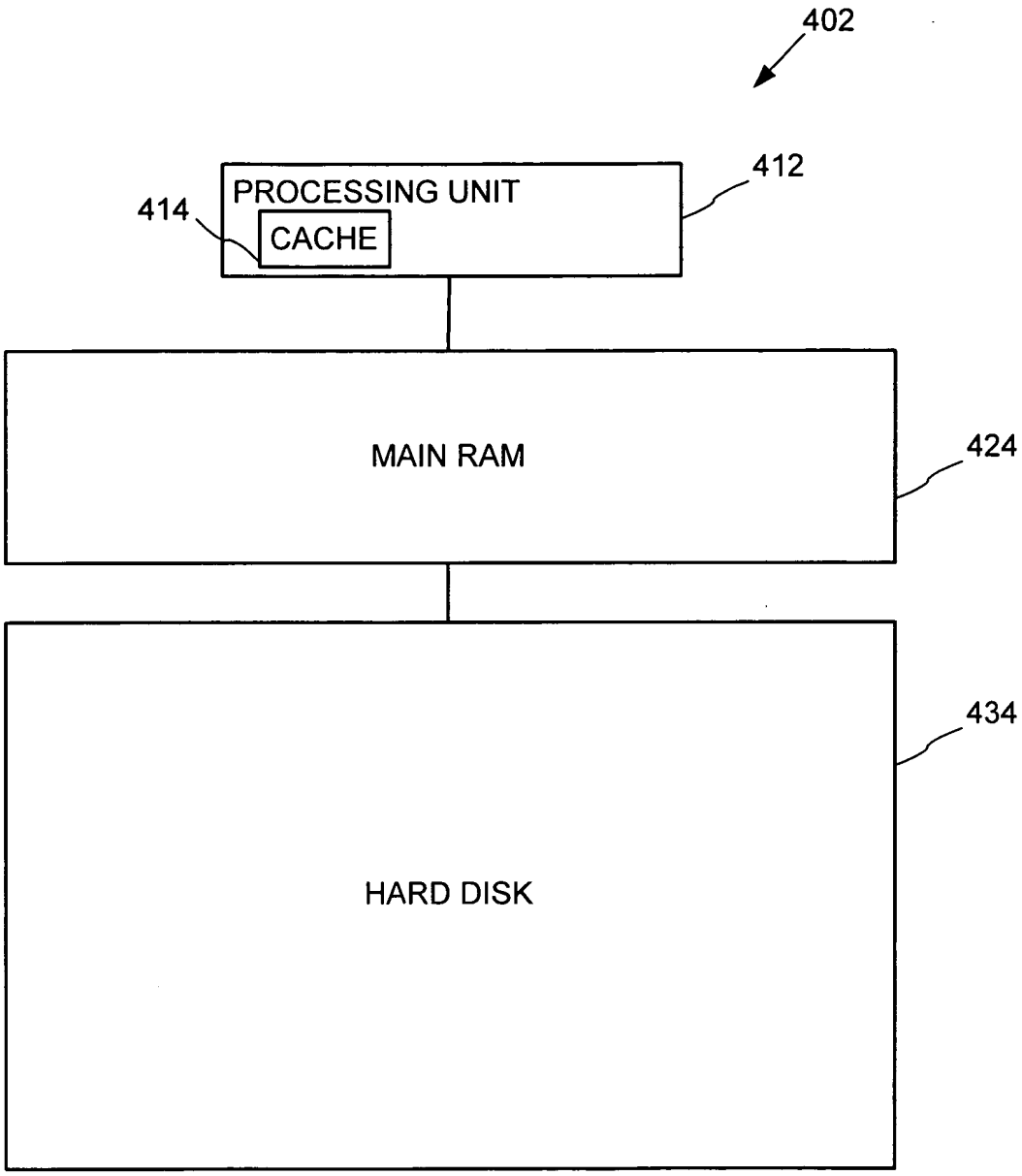


FIG. 5

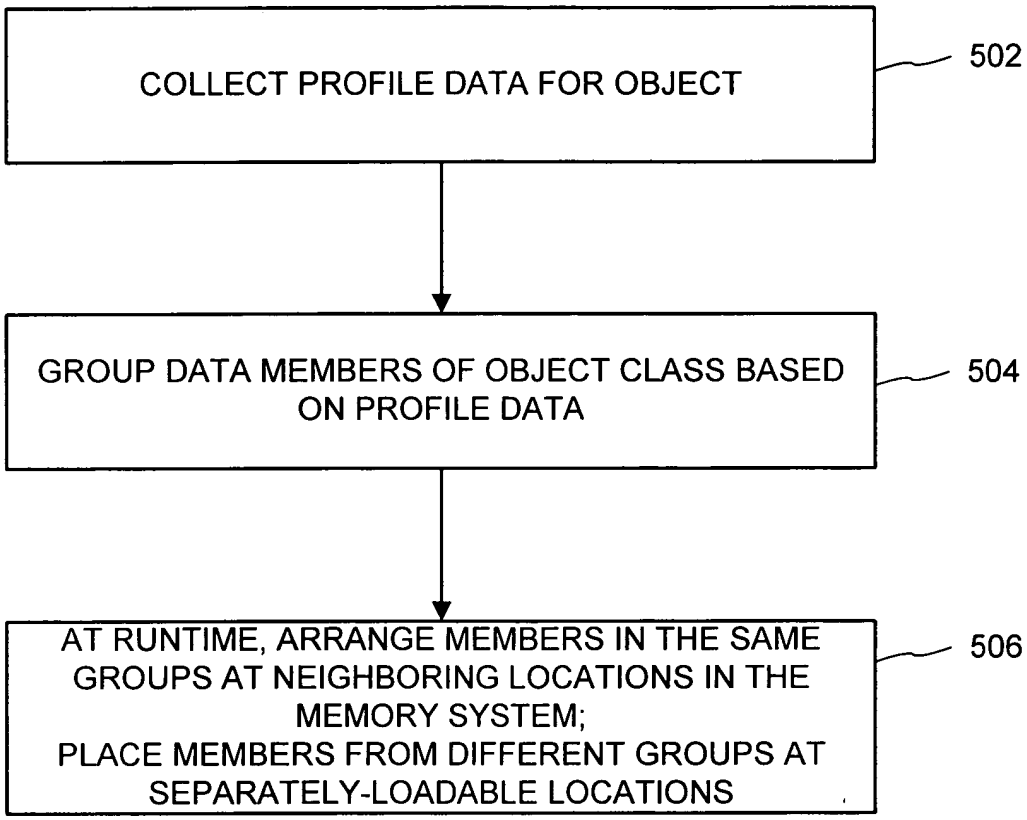

COLLECT PROFILE DATA FOR OBJECT
502
GROUP DATA MEMBERS OF OBJECT CLASS BASED ON PROFILE DATA
504
AT RUNTIME, ARRANGE MEMBERS IN THE SAME GROUPS AT NEIGHBORING LOCATIONS IN THE MEMORY SYSTEM;
PLACE MEMBERS FROM DIFFERENT GROUPS AT SEPARATELY-LOADABLE LOCATIONS
506

FIG. 6

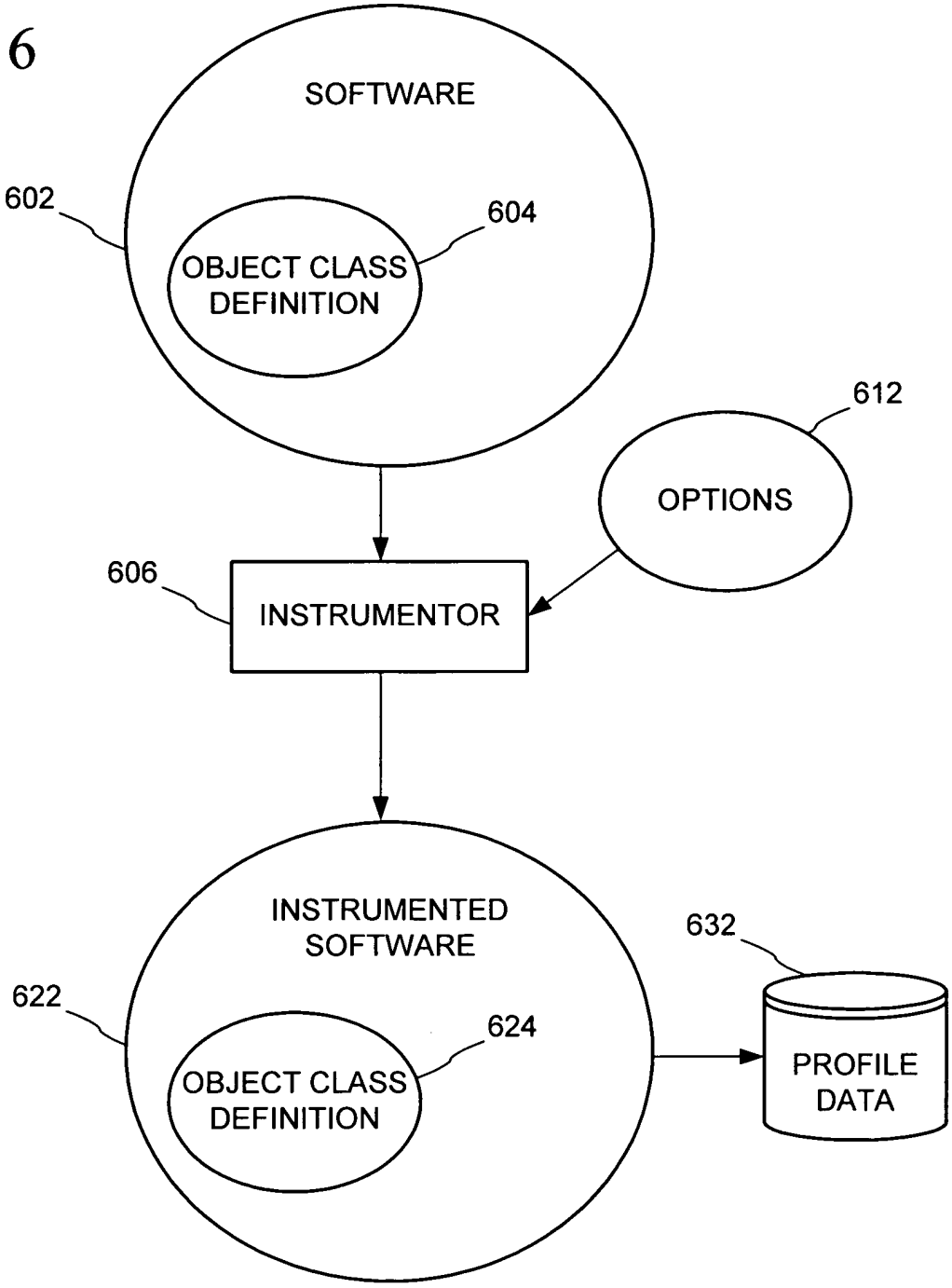

SOFTWARE
602
604
OBJECT CLASS DEFINITION
612
OPTIONS
606
INSTRUMENTOR
INSTRUMENTED SOFTWARE
632
622
624
OBJECT CLASS DEFINITION
PROFILE DATA

FIG. 7

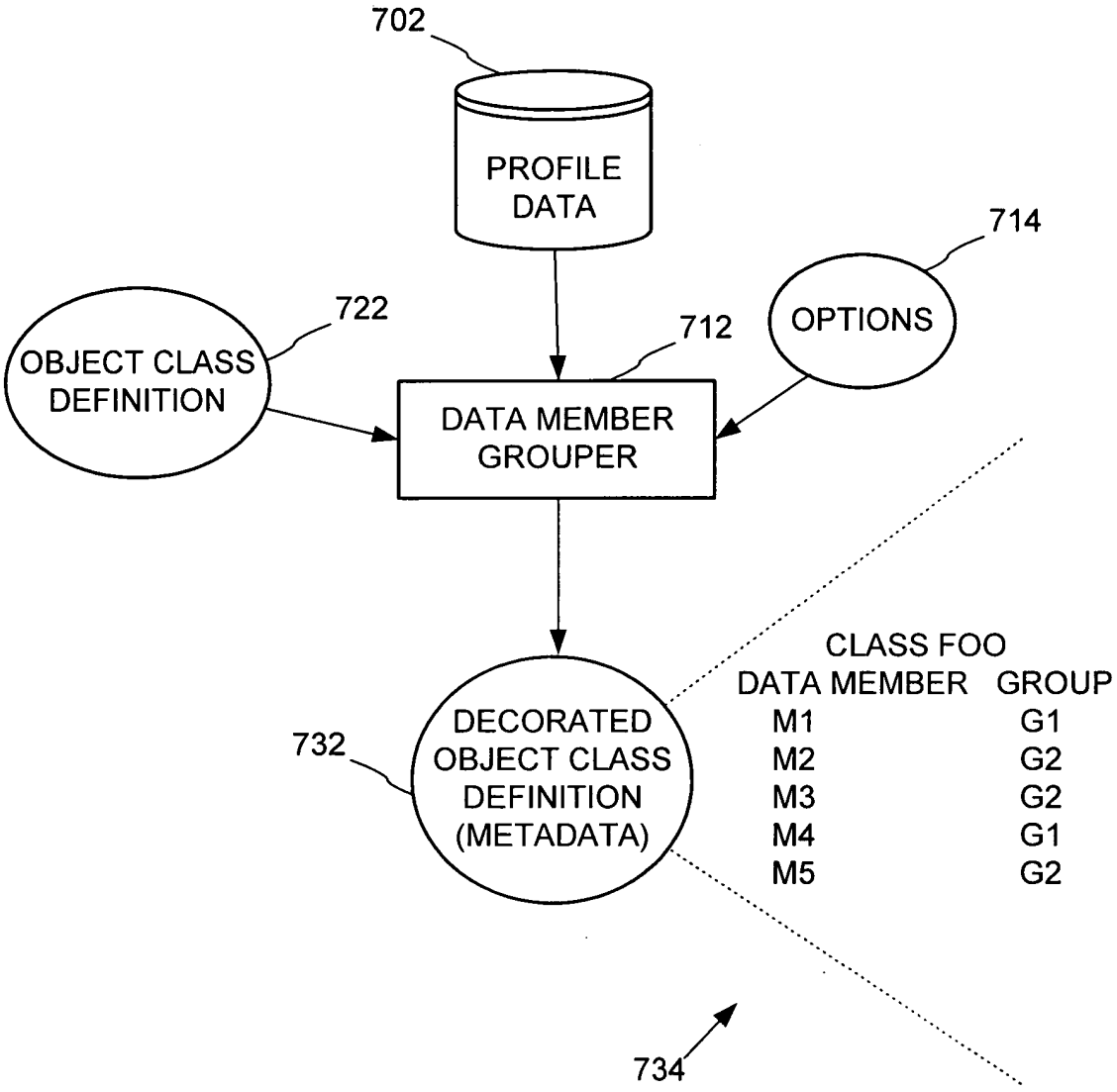

702
PROFILE DATA
714
OPTIONS
722
OBJECT CLASS DEFINITION
712
DATA MEMBER GROUPER
CLASS FOO
DATA MEMBER GROUP
M1 G1
M2 G2
M3 G2
M4 G1
M5 G2
732
DECORATED OBJECT CLASS DEFINITION (METADATA)
734

FIG. 8

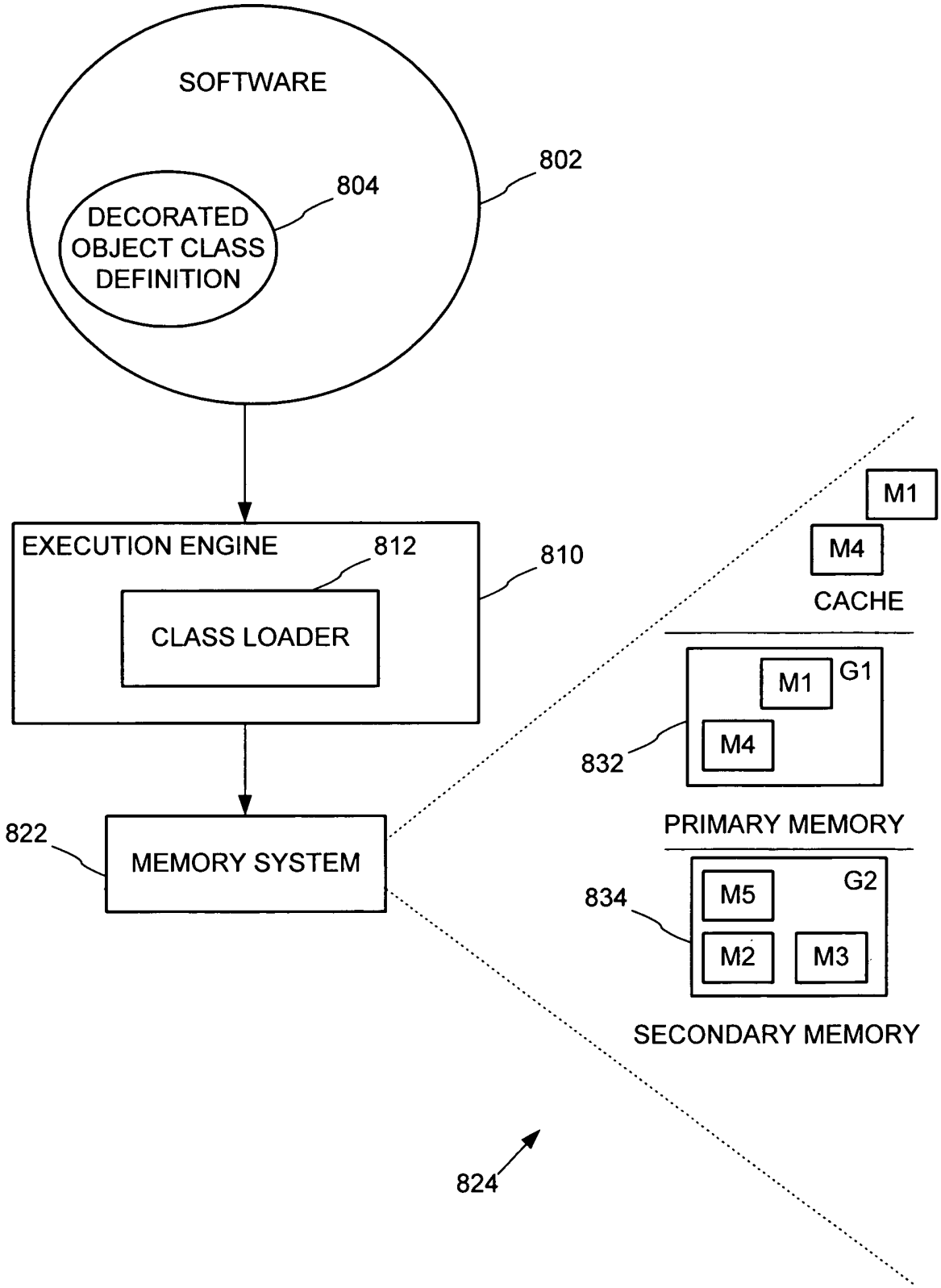

SOFTWARE
802
804
DECORATED OBJECT CLASS DEFINITION
EXECUTION ENGINE
812
810
CLASS LOADER
822
MEMORY SYSTEM
M1
M4
CACHE
M1
G1
M4
832
PRIMARY MEMORY
M5
G2
834
M2
M3
SECONDARY MEMORY
824

FIG. 9

```
void SaveHitCountsAndPossiblyFlushRecord ()
{
// H = hits so far (overall count)
//H1 = previous value of H
// L = Hits observed during last segment
//          (e.g., method execution)
  if ((H + L) < H)  // overflow this time
    FlushRecord () ;
  else
  {  H+=L ;  //bump overall count by current counts
    L = 0 ;
    // will we overflow next time? (linear forecast)
    if ( ((H - H1) + H) < H)
      FlushRecord  ();
    else
      H1 = H ; // remember current count
  }
}

void FlushRecord ()
{  Write (H, fieldName, className) ;
  H1 = 0 ; // zero out the previous count
  H = L;
  L = 0 ;
}
```

FIG. 10

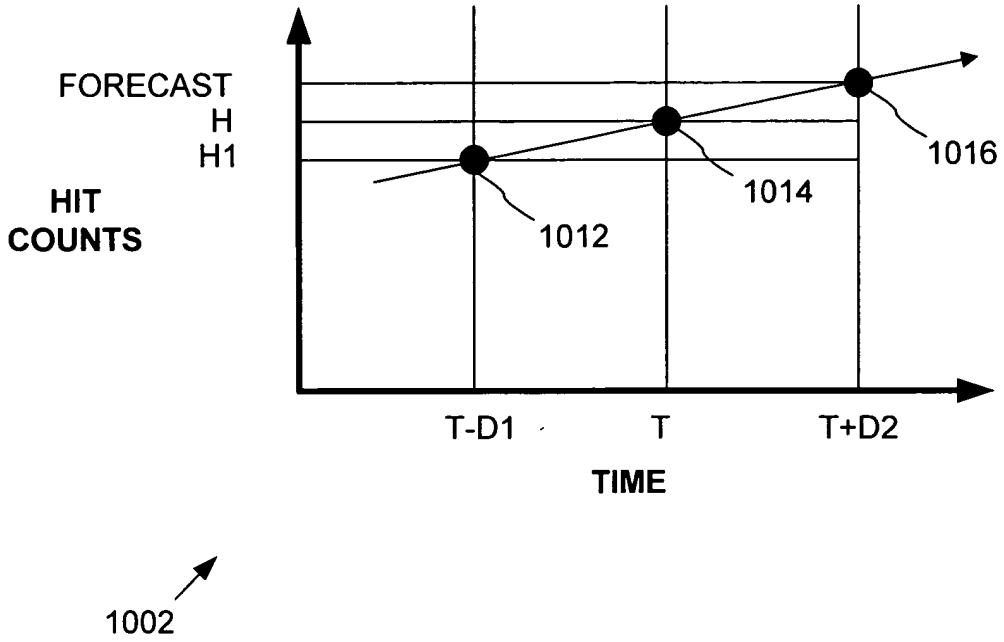


FIG. 11

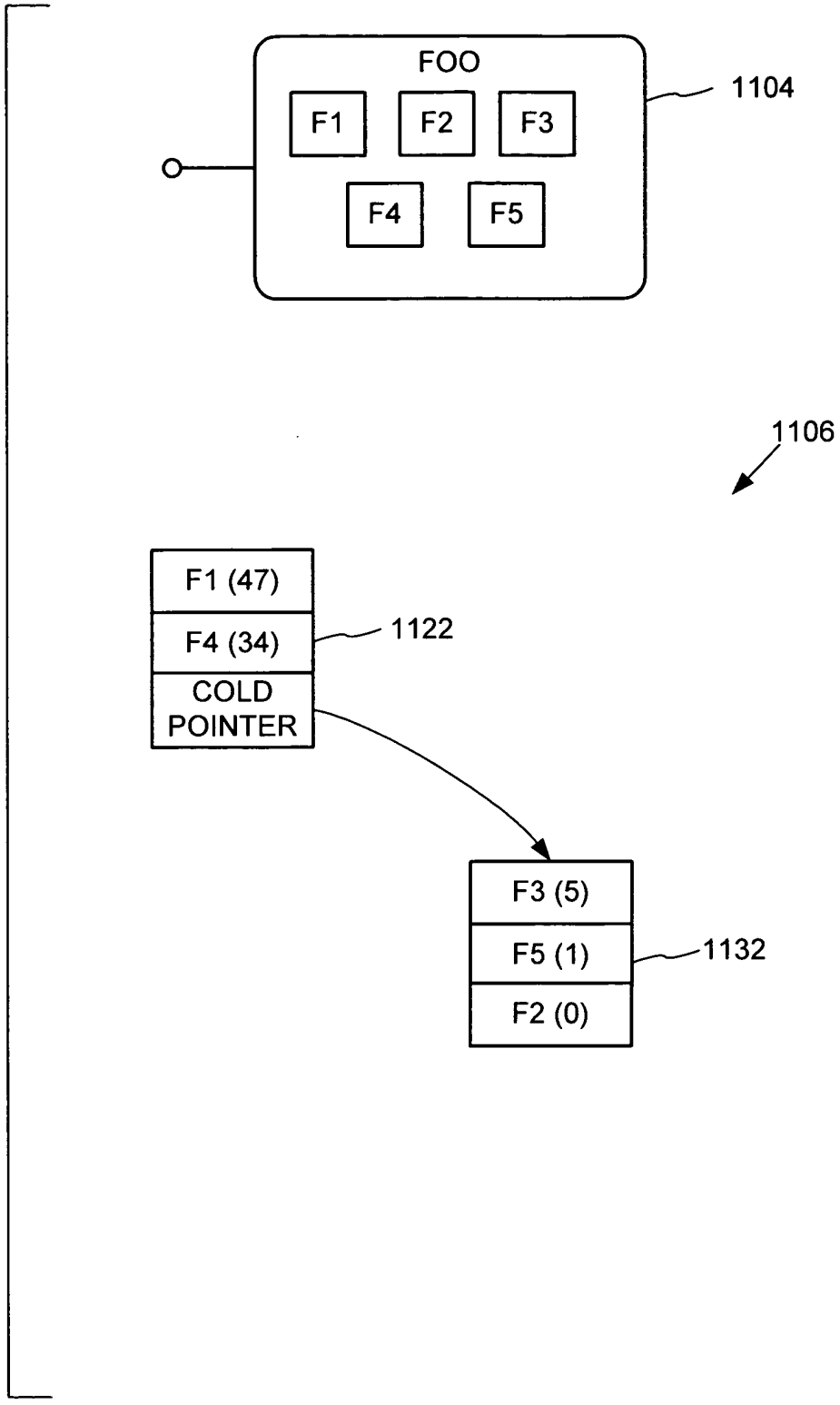

FOO
F1
F2
F3
F4
F5
1104
1106
F1 (47)
F4 (34)
COLD POINTER
1122
F3 (5)
F5 (1)
F2 (0)
1132

FIG. 12

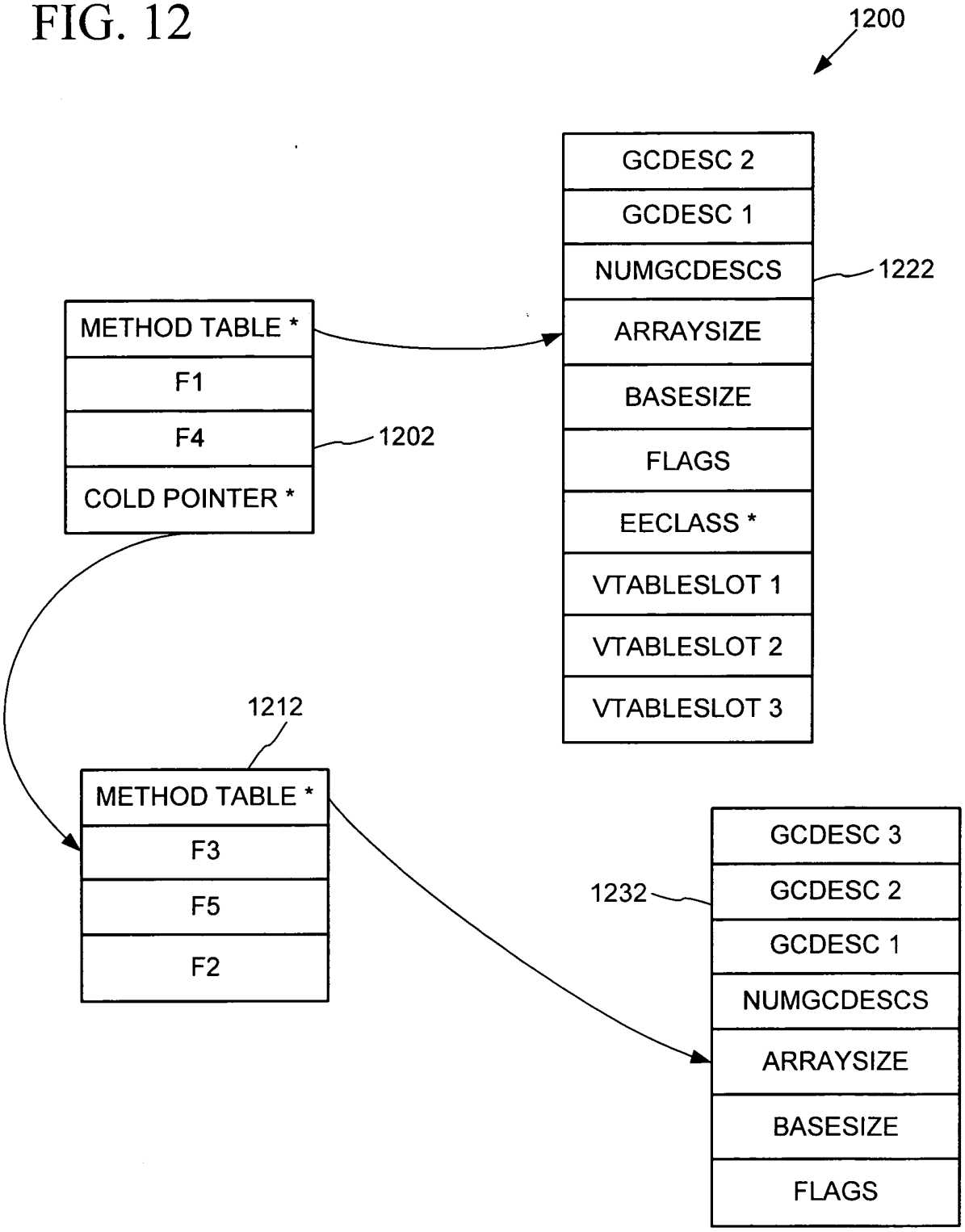

1200
GCDESC 2
GCDESC 1
NUMGCDESCS
1222
ARRAYSIZE
BASESIZE
FLAGS
EECLASS *
VTABLESLOT 1
VTABLESLOT 2
VTABLESLOT 3
METHOD TABLE *
F1
F4
1202
COLD POINTER *
1212
METHOD TABLE *
F3
F5
F2
GCDESC 3
1232
GCDESC 2
GCDESC 1
NUMGCDESCS
ARRAYSIZE
BASESIZE
FLAGS

FIG. 13

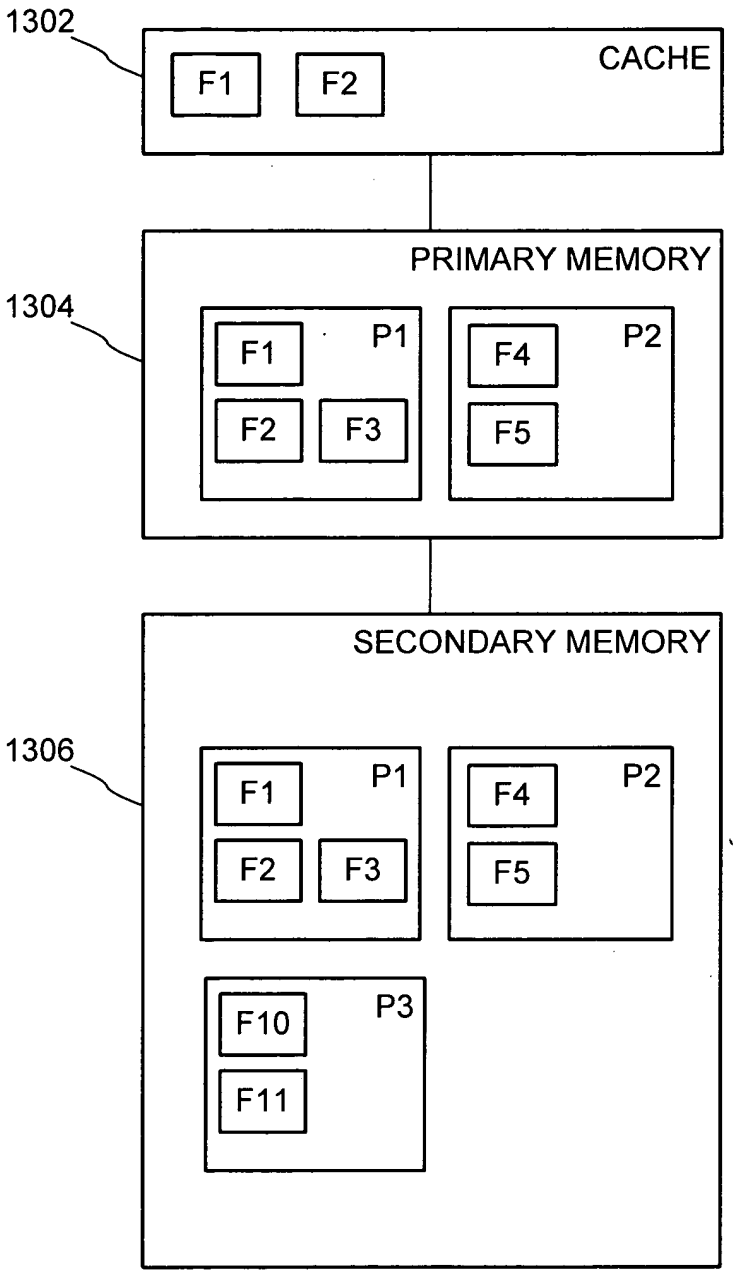

1302
CACHE
F1
F2
PRIMARY MEMORY
1304
P1
F1
F2
F3
P2
F4
F5
SECONDARY MEMORY
1306
P1
F1
F2
F3
P2
F4
F5
P3
F10
F11

FIG. 14

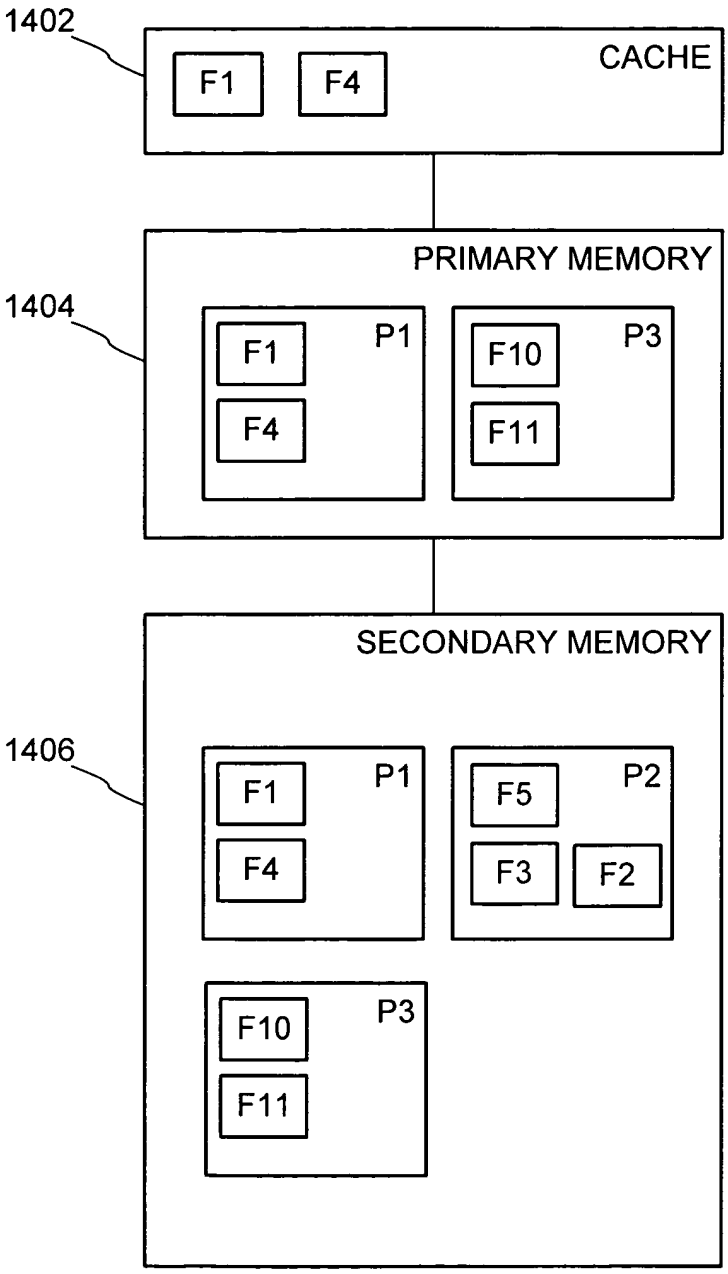

1402
CACHE
F1
F4
PRIMARY MEMORY
1404
P1
F1
F4
P3
F10
F11
SECONDARY MEMORY
1406
P1
F1
F4
P2
F5
F3
F2
P3
F10
F11

FIG. 15

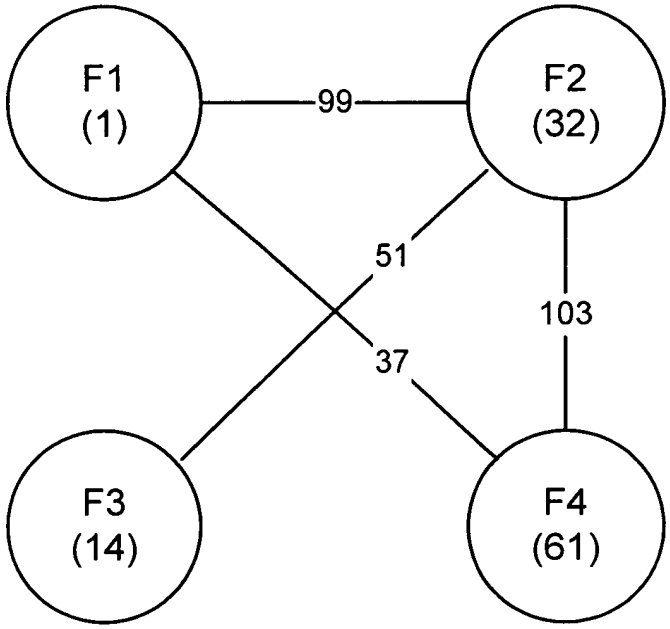


FIG. 16

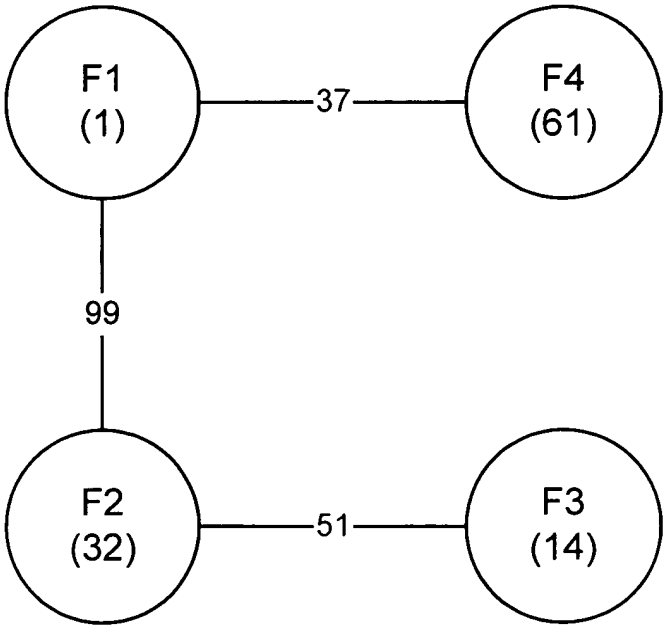
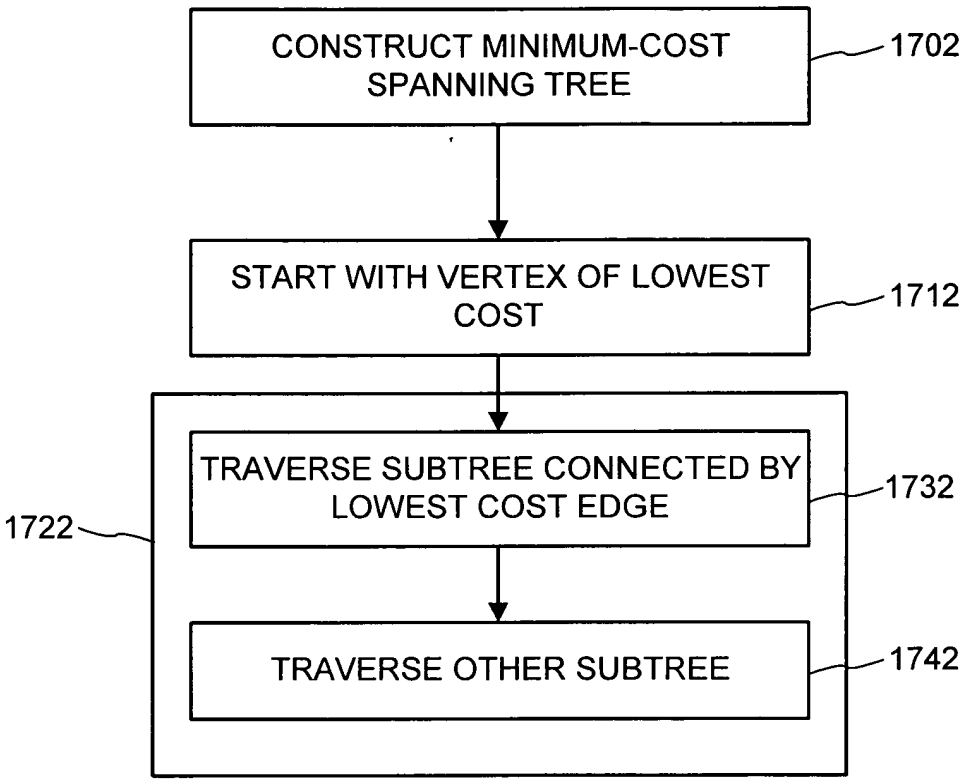


FIG. 17


CONSTRUCT MINIMUM-COST SPANNING TREE
1702
START WITH VERTEX OF LOWEST COST
1712
TRAVERSE SUBTREE CONNECTED BY LOWEST COST EDGE
1732
1722
TRAVERSE OTHER SUBTREE
1742